PROJECT DESCRIPTION

In this project, you will represent polynomials using linked lists, and write methods for symbolic addition and multiplication of polynomials. The interface for the Class Polynomial is provided below:

```java
public class Polynomial {
    public static Polynomial add(Polynomial p1, Polynomial p2){}
    public static Polynomial multiply(Polynomial p1, Polynomial p2){}

    private class PolyNode{ // Each node
        double coeff;
        int exp;
        private PolyNode next;
        public PolyNode(double c, int e, PolyNode n){
            coeff = c; exp = e; next = n;
        }
    }
    private PolyNode poly;
    public Polynomial(String str){ // PRE: str structurally correct, and in sorted order }
    public void print(){ }
}
```

The following snippet of code illustrates the use of the Polynomial class:

```java
Polynomial p1 = new Polynomial("5 3 3 1 2 0");
Polynomial p2 = new Polynomial("6 3 2 1");
Polynomial.add(p1, p2).print();
Polynomial p3 = Polynomial.multiply(p1, p2);
p3.print();
```

The second and third lines define the polynomials \( p_1 = (5x^3 + 3x + 2) \) and \( p_2 = (6x^3 + 2x) \) respectively. The fourth line asks you to print the polynomial \( p_1 + p_2 \), which is \( 11x^3 + 5x + 2 \). The fifth line sets \( p_3 \) to the polynomial \( p_1 \times p_2 \), or \( (30x^6 + 28x^4 + 12x^3 + 6x^2 + 4x) \); the sixth line prints \( p_3 \). The expected output of this piece of code is

\[
11 \times^3 + 5 \times + 2 \\
30 \times^6 + 28 \times^4 + 12 \times^3 + 6 \times^2 + 4 \times
\]

The formatting of the doubles does not matter. The following is also valid output:

\[
11.0 \times^3 + 5.0 \times + 2.0 \\
30.0 \times^6 + 28.0 \times^4 + 12.0 \times^3 + 6.0 \times^2 + 4.0 \times
\]

For this assignment you are to complete the implementation of the Polynomial class whose interface is provided above. Specifically,

1) Complete the implementation of the constructor. This accepts as input a String that lists the coefficients and exponents of the terms of the polynomial. You may assume (as a precondition) that this string is (i) structurally correct (i.e., it contains an even number of numbers, of which the first, third, fifth,... are doubles and the second, fourth, sixth, ... are ints); and (ii) lists the terms in decreasing order of exponents.

2) Implement the print method. This should produce output as illustrated above, with the terms of a polynomial being listed in decreasing order of exponent.

3) Implement the add and multiply methods, which should behave as indicated in the example.
**Grading Rubric**

- (10 points) Correct implementation of the constructor
- (15 points) Correct implementation of the add method
- (15 points) Correct implementation of the multiply method
- (5 points) Correct implementation of the print method
- (10 points) Efficiency of the constructor
- (15 points) Efficiency of the add method
- (15 points) Efficiency of the multiply method
- (5 points) Efficiency of the print method
- (10 points) Evaluate your code and provide the big-O analysis of these four methods in a comment above each one.

Note that credit will be awarded for efficiency in implementation. Make your implementation efficient in terms of both run-time and memory usage.

For the last item in the list, add the big-O complexity in a comment above each method. If your analysis is correct then you will receive full credit for this, even if you had points taken off for efficiency. The following is an example of commenting the complexity:

```java
/**
   * Runtime: O(n log n)
   * Memory: O(n^2)
   */
public void foobar() {
```

**Submission Instructions**

Upload all your source code in a `.zip` file to Sakai. You are responsible for ensuring that your program compiles and functions properly. Any non-functioning program will receive a zero.

**Honor Code**

Please review the honor code description from the course syllabus. No collaboration (with anyone) is permitted in assignments. Collaboration in assignments, or the use of code not the students own, constitutes an honor code violation. Any violation will be reported to the Student Attorney General.

**Appendix A: Required Methods**

Your `Polynomial` class should support the following constructor and operations:

```java
public static Polynomial add(Polynomial p1, Polynomial p2);
public static Polynomial multiply(Polynomial p1, Polynomial p2);
public Polynomial(String st);
public void print();
```

The `Polynomial` class skeleton is provided on the following page, so you can copy and paste it into your editor to begin working on it.

**Appendix B: Approved Libraries**

Do not use any in-built Java collection implementation (e.g., List, Queue, Map, Set, etc) from the `java.util.*` API. You must implement your entire program from scratch.

The only exception, however, is the `java.util.Scanner` class. You may use this if you want and know how.
public class Polynomial {
    public static Polynomial add(Polynomial p1, Polynomial p2) {
        // TODO: Implement me
    }

    public static Polynomial multiply(Polynomial p1, Polynomial p2) {
        // TODO: Implement me
    }

    private class PolyNode {
        double coeff;
        int exp;
        private PolyNode next;

        private PolyNode(double c, int e, PolyNode n) {
            coeff = c; exp = e; next = n;
        }
    }

    private PolyNode polyNode;

    public Polynomial(String str) {
        // TODO: Implement me
    }

    public void print() {
        // TODO: Implement me
    }
}